International Journal of Electronics, Communication & Instrumentation Engineering Research and Development (IJECIERD)
ISSN (P): 2249–684X; ISSN (E): 2249–7951
Vol. 12, Issue 1, Jun 2022, 37–46

TRANS STELLAR
•Journal Publications •Research Consultancy

EFFECT OF FREQUENCY SCALING ON POWER CONSUMPTION IN EMBEDDED SYSTEMS

SANDEEP S CHAPALKAR & K. KARIBASAPPA

Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India

ABSTRACT

Optimization of power consumption in a real time embedded systems is a critical design parameter. As most of low power embedded systems are battery operated and have very limited source of power. Therefore the optimization of power utilization is discussed in this paper establishing its relationship with execution frequency of core processing unit in multitasking environment. The number of processes required to be executed in multitasking embedded systems may vary with time for an application, and hence the dynamic power consumed also vary accordingly. Each process in real time are defined with respective absolute deadlines and their relative deadlines depend on the number of processes schedules at a time. The relative deadline parameter can be used to estimate the required operating frequency of processing unit and can be scaled accordingly to optimize the power consumption. The relative deadlines needs to be less than the absolute deadline for any process to complete the task successfully. Therefore the operating frequency can be dynamically scaled such that the relative deadlines are just within absolute deadlines.

The hard-real time systems are designed to meet the deadline of all processes under maximum process load, hence the operating frequency of the processing unit is chosen much higher to complete make sure all the processes within deadline under peak load condition. However, scaling down the frequency under other conditions will improve the thermal stability and also optimize the power consumption.

The analysis of frequency requirement is done for two most widely used scheduling methods, Earliest Deadline First (EDF) and Rate Monotonic (RM) scheduling algorithms for comparison.

KEYWORDS: Power consumption, Process Load, Dynamic load, Multitasking, Frequency scaling, Relative Deadline, Absolute Deadline

Received: Feb 12, 2022; **Accepted**: Mar 02, 2022; **Published**: Mar 12, 2022; **Paper Id.**: IJECIERDJUN20223

Original Article

INTRODUCTION

The performance of multitasking real time systems are measured not only based on completion of the jobs assigned but also on the time consumed to complete the jobs successfully. The system is said to be performing better if it can complete all the assigned jobs or tasks within a defined deadline with minimum power consumption. The aggressive power management techniques are very important in embedded systems specially those which are battery operated.

The design related issues are affected by the time constraints of a system. The real time performance of system depends on successful execution of processes within relative deadlines which are decided dynamically based on the process load. The hard real time systems are designed with sufficiently high frequency processor to meet the deadlines in worst case scenario when all processes are active and scheduled in the system. However,

most of the time systems will not be executing all processes, in such scenarios deadlines can be met even by operating at much lower frequencies and the by saving the power under such circumstances. There by frequency scaling can be implemented to optimize the power consumption. The scaling factor is calculated based on the relative deadlines of scheduled processes.

The general equations related to calculation of various parameters such as absolute deadline, relative deadline, frequency scaling factor are discussed and then the comparative analysis is done using EDF and RM scheduling algorithms. The few dynamic parameters are logically assumed considering their relationship and impact on overall task execution to simulate the real time conditions.

RELATED WORK

Frequency Scaling Factor

A real time embedded system with N number of processes may execute only selected processes under different conditions. Considering X is a count of maximum processes that are scheduled in system under peak load condition. The processing unit operates at highest possible frequency to achieve T_{RD} (Relative Deadline) less than T_{AD} (Absolute Deadline) for all the processes. However if system is working with less number of processes then frequency can be scaled to lower level which in turn reduces the power consumption. If in any case T_{RD} of process exceeds its absolute deadline T_{AD}, then frequency of operation must be scaled up and if T_{RD} is much lesser than T_{AD} then operating frequency needs to be scaled down. The frequency scaling is proportional to difference between T_{RD} and T_{AD} of last process in the process queue.

$$f \alpha (T_{RD} - T_{AD}) \quad (1)$$

If $(T_{RD} - T_{AD})$ is positive, then frequency is scaled up and if it's negative then frequency is scaled down. The scaling factor is calculated as the ratio of difference between the relative and absolute deadline to relative deadline achieved. In multitasking, the process having highest scaling factor is considered.

$$k = (T_D - T_{RD}) / T_{RD} \quad (2)$$

Hence, the new required frequency of operation f_R required to meet the deadline is given by equation 3.

$$f_C = k * f \quad (3)$$

The frequency scaling algorithm involves following steps.

- Identify the processes completing the execution within scheduling period but missing the deadline.
- If all scheduled processes are meeting the deadline and some processes are left out without getting even single slot then frequency is arbitrarily scaled up by 25 % and checked for updated status.
- The frequency scaling factor is calculated for all processes completing the execution but missing the deadline.
- Select the highest factor and scale the frequency accordingly.
- Check if all the processes are meeting the deadline using revised frequency.
- If 'YES' then same frequency is maintained for application, otherwise the cycle repeats from step i till all processes meet the deadline requirements.
- The lowest operating frequency is computed such that at least one process violates the deadline below that

frequency [1].

The system requires and additional programs to monitor the change in turnaround time due to dynamic variation in number of processes and to calculate the scaling proportion of operating frequency. These processes are to be active throughout the operating period of the system. The additional overhead introduced due to these background processes to monitor this operation also should be considered during design cycle.

Frequency Multiplier Techniques

The frequency multiplier techniques can be used to scale the operating frequency of processing unit if it can be integrated as a module within the processor. There are various techniques invented overt the period are listed below.

- There are many variable frequency multipliers based on energy distributed on harmonics and input multiplier [2]. These multiplier circuits can be used to scale the frequency as and when required.
- Phase Locked Loop can be configured as frequency multiplier which used multiphase voltage controlled oscillator [3].
- Another technique to scale the frequency is sinusoidal frequency doublers characterized by tuned LC circuits [4].
- The frequency and voltage of processor can be dynamically changed based on processor utilization factor to reduce the overall power consumption [5].
- A neural network model technique also can be used to predict the future load on the system and decide the scaling factor in advance [6] [7].
- Used Driven Frequency Scaling is another efficient and usage based customized frequency scaling technique [8].
- Learning direct frequency scaling model used Counter Propagation Network (CPN) to predict the behavior of task and compute corresponding frequency required [9].

Power Performance Impact in Multitasking Systems

The performance of multitasking real time systems are measured not only based on completion of the jobs assigned but also on the time consumed to complete the jobs. The system is said to be performing better if it can complete all the assigned jobs or tasks within a defined deadline with minimum power consumption. The aggressive power management techniques are very important in embedded systems specially those which are battery operated. Programmable clock gating and DVFS are extensively used for power management in SoC technology [10]. Therefore the operating frequency of the processor should be scalable to achieve best performance. In case of multitasking systems deadline partitioning with fair DVFS (DPF-DVFS) can be used to schedule processes optimally [11]. The real time scheduling in multiprocessor systems can also be achieved by deadline partitioning [12] [13] [14] [15]. The optimal energy efficient techniques for real time systems having periodic tasks are achieved but there are no optimal algorithm for systems having sporadic tasks.

Process scheduling in modern technologies like cloud server poses a greater challenge as the heterogeneity is involved in both resources used as well as processes to be serviced [16]. Heterogeneous resources are managed by Virtual Machine Monitor (VMM) for efficient utilization and optimal energy consumption [17]. The heterogeneous design of

processor with multi-core technology leads to load distribution concept to optimize the average energy consumed by each core [18]. Partitioning the total task set into subset and permanently binding them to a particular core with separate run queue is one method can be implemented in systems having fixed or deterministic tasks set [19][20]. The optimal distribution of load can be formulated by heuristic method load distribution and assignment method using combination of DVFS and DFS with low latency sleep states [21].

The voltage required for the stable operation of processing unit depends on operating frequency. Hence operating voltage can be scaled proportional to operating frequency with consideration of minimum voltage requirements. There are two basic components viz. static and dynamic power which decides the overall power consumption of processing unit. The static power consumption is basically due to leakage current in various circuits in idle state of processor and dynamic power is based on clocking frequency while processor is actively execution a process. Due to static power consumption and asymptotic execution time the software code execution shows convex energy behavior, i.e., there exists an optimal CPU frequency at which energy consumption is minimal. Leakage current is playing more important role as transistor sizes have become smaller and threshold voltage levels lower. A decade ago, dynamic power accounted for approximately two-thirds of the total chip power. The processes associated with tasks consists for set of instructions to achieve the required operation. These instructions can be broadly classified as memory bound, I/O bound and CPU bound.

Memory bound instructions are used to access (read / write) to and from primary memory of the system. As operating speed of memories are much slower compared to processors, the scheduler may force the particular process to suspended state and next process starts executing. However if the process being executed is of higher priority compared to other processes then it will not release the resources allocated. This may lead to reduction in the overall performance of the device. In such cases the process hold the processor till it retrieve the required data from the memory wasting precious time of processing unit. Therefore it's desired to reduce memory bound instructions as much as possible in case of high performance systems. To achieve this processors support large set of registers and cache memory architecture.

I/O bound instructions deals with exchange of data with peripheral inputs or output ports. The response time of these instructions depends on the response ability of the device it is trying to communicate. In some cases if the data is expected over network or user interfaces then this operation will take considerably large time. However for better interactive systems it is necessary to have I/O bound operations to provide proper user interaction and operational feedback.

CPU bound instructions are completely depends on processor for completion of the task and hence are executed much faster than other two types of instructions. All the data processing and manipulations are done by such instructions and they form large part of program in processing oriented applications. Whereas these instructions are limited in user or network interfacing programs where in other two type of instructions dominate.

The power efficiency of system can be improved by reducing the dynamic power consumption by operating at optimal frequency to meet the required deadlines. This also improves thermal efficiency as voltage requirement reduces with respect to frequency. DFS reduces the number of instructions executed by the CPU in unit time and hence normally used when programs contain mostly an IO bound instructions.

The same phenomenon of DFS [21] can be used to optimize power consumption in case of multitasking systems by scaling the frequency to just meet the required deadlines. As the number of processes in the execution queue increases

then operating frequency is also scaled accordingly to meet the deadlines. If the current frequency itself can complete the executions of all processes within deadlines, then frequency can be maintained same and can be scaled down to save more power. As completion of process execution much before its deadline does not provide any considerable incentives in terms of system performance.

The "Advanced Configuration and Power Interface (ACPI)", provides an open standard operating system to discover and configure hardware components of computer [22]. In most of modern CPU "P" – states are defined which allow clock rate reduction and "T" states (throttling states) which allow to throttle down actual clock rate. Using ACPI power management comes under OS instead of BIOS which relied on platform specific firmware to determine power management and configuration policies.

The MATLAB programs are designed to calculate relative deadline with respect to process ID and plot the variation. As general computer systems or laptops does not allow user to change the operating frequency of the processor directly using APIs due to safety reasons. However, the requirement of scaling can be analyzed based on variation of relative deadlines. If all processes are completed well within the required deadline then frequency can be scaled down and if any process is missing the deadline then frequency of operation need to be scaled up proportionately.

Simulation of Frequency Scaling Algorithm

The frequency scaling is simulated in MATLAB considering five set of processes with both EDF and RM scheduling principles. Each process is characterized by its execution time (Abs_Te), period of the process (Period_P) and their respective absolute deadlines (Abs_Deadline). Then based on algorithms explained in previous sections the execution queue is created for both scheduling policies. Three scaling factors are considered for frequency to illustrate the variation in relative deadlines. Algorithm of program to illustrate frequency scaling impact on multitasking system is given below.

- Initialize execution time, absolute deadline and period of process.
- Priority list are created for EDF and RM scheduling policies and used as execution queue sequence.
- Frequency scaling factors are defined as 1.0, 1.5 & 2.0.
- Relative deadlines are calculated for different scaling factors.
- The figure is plotted to indicate variation of relative deadlines of process for all scaling factor and compared to identify the factor satisfying all deadlines.

The deadlines defined for the processes are 6, 4, 10, 16, and 12 for five processes respectively. Each process is given twice as much time as its execution time as deadline. In soft and firm real time systems the deadlines are slightly relaxed to reduce the cost and power consumption. The relative deadline equation defined in chapter two is used for the calculations and plotted on the figure for both EDF and RM scheduler with three different frequency scaling factors as represented by the legends in each figure.

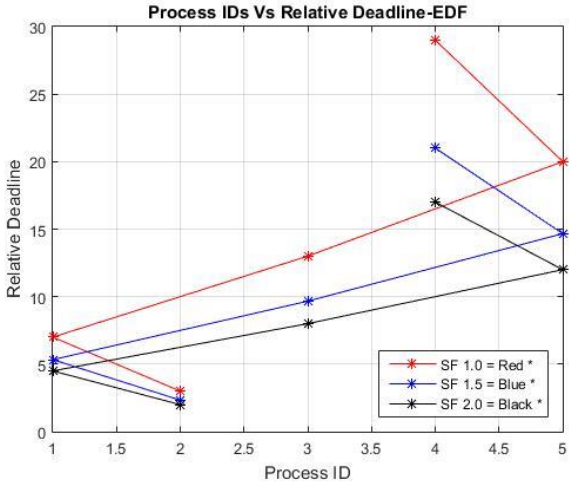

Figure 1: Impact of Frequency Scaling in EDF

In EDF scheduling, the order of process queue is 2, 1, 3, 5, and 4 assuming all processes are ready for execution. Following observations can be depicted from the variation of relative deadline plotted.

- Scaling factor 1.0: Process 2 meets the deadline where all other processes 1, 3, 4, 5 are missing the required deadline.
- Scaling factor 1.5: Process 1, 2, 3, 4 meet the deadline whereas process 5 misses required deadline.
- Scaling factor 2.0: Process 1, 2, 3, 4 meet the deadline whereas process 5 misses required deadline.

Therefore, it can be concluded that if the processes are to be scheduled using EDF policy then system required and CPU with higher operating frequency. The higher frequency also indicates higher power consumption. In real time systems specially operating on battery backup it's very important to optimize the consumption of power. This analysis will be of great help in such cases to select the appropriate scheduling policies.

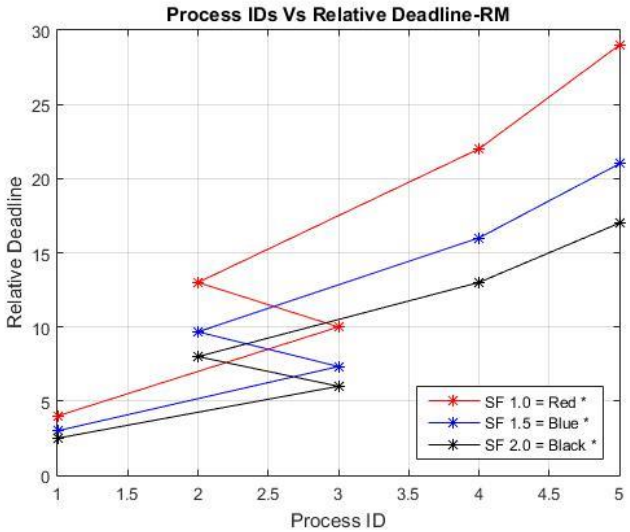

Figure 2: Impact of Frequency Scaling in RM

The order of scheduling in RM policy is 1, 3, 2, 4, 5 with all processing are in ready queue of scheduler. The following observations can be concluded by the figure 2.

- Scaling factor 1.0: Process 1 meets the deadline where all other processes 1, 3, 4, 5 are missing the required deadline.
- Scaling factor 1.5: Process 1, 3, 4 meet the deadline whereas process 2, 5 misses required deadline.
- Scaling factor 2.0: Process 1, 3, 4 meet the deadline whereas process 2, 5 misses required deadline.

The comparison of the two scheduling policies for the set of processes considered reveals that even if the operating frequency is scaled by the factor of 2.0, two processes miss deadline in case of RM scheduling whereas only 1 process misses the deadline in case of EDF policy. If the hardware limitations are imposed and higher frequency CPU is not allowed, then EDF scheduling is efficient for the system.

Estimation of execution time for processes can be done in two different methods. Which helps in designing the system with appropriate CPU and hardware resources. As already discussed, if the system is hard real time then system is designed assuming maximum number of processes in ready state. In case of firm and soft real time systems the design is done with consideration of average number of processes or total periodic processes. There are different methods to estimate the execution time and resource requirements. Two simple and effective methods applicable in case of commercial real time systems are discussed below.

In first method, all the programs are converted to equivalent assembly language program and then clock cycles required for each instruction is verified. The total number of clock cycles required for program is calculated and multiplied with clock period to get the execution time of program. Once the execution time is calculated for all program's execution time is compared with required deadlines. If any program fails to meet the required deadline then CPU can be upgrade to higher operating frequency to meet the requirement. However, this method becomes complicated for complex programs and if many sub functions are included is the system operation.

$$T = \sum_{i=0}^{i=n} CPI(i) * T_{cycle} \quad (4)$$

T = Time required for execution of program

n = Number of instructions

CPI = Machine cycle per instruction for i_{th} instruction

T_{cycle} = Time period of one machine cycle

Machine cycle period varies with respect to frequency 'f' and clock cycles per machine cycles.

The second method is based on estimation on standard processing unit and then comparing the performance with target hardware designed for the system. In this method each program is executed on standard CPU and execution times are tabulated and then based on these values the actual execution on target hardware is calculated. This method is very simple and used only when the target hardware or CPU is not available during design cycle of system. Mostly used in hard real time system for which hardware are specially designed as per requirement of the system.

The third method is the directly execute the programs on target hardware and observe the execution time under

different scenarios and calculate average execution time. This process requires hardware to be known while designing the system. This method is most effective and efficient in firm and soft real time system design.

The third method explained is used in proposed work to calculate the execution time of each process. The time required in case of single process execution and time required when process is invoked as function in another process both are calculated on Raspberry Pi 3b+ operating at 700MHz standard frequency.

The flow diagram of implementing dynamic frequency adaptation in multitasking environment is shown below, the scheduling policies are indicated with common block diagram as they only modify the process execution queue. Whereas the entire system operation and decision-making process remains same for any scheduling policy. Hence the technique proposed can be used with implementation of any scheduling policies and not limited to the EDF and RM policies.

REFERENCES

1. *Youngsoo Shin, Kiyoung Chois, and Takayasu Sakurait, "Power Optimization of Real-Time Embedded Systems on Variable Speed Processors", 0-7803-6445-7/00 2000 IEEE.*
2. *Didier Quievy, Francis Desjouis "Frequency Multiplier with Programmable Order of Multiplication", Patent Number: 4,967,160, Date of Patent: Oct. 30, 1990*
3. *William S. JenningsCheck, "Clock adapter using a PLL configured as a frequency multiplier with a non-integer feedback driver", Patent Number: 5,059,924, Date of Patent: Oct. 22, 1991.*
4. *Bancha Burapattanasiri "Sinusoidal Frequency Doublers Circuit with Low Voltage + 1. 5 Volt CMOS Inverter", International Journal of Computer Science and Information Security, Vol. 6, No. 3, 2009*
5. *Chunling Hu, Jack Liu," Method and Apparatus for Dynamic Voltage and Frequency Scaling", Patent No.: US 7,730,340 B2, Date of Patent: Jun. 1, 2010.*
6. *Anirban Lahiri, Nagaraju Bussa, Pawan Saraswat, "A Neural Network Approach To Dynamic Frequency Scaling", 15th International Conference on Advanced Computing and Communications, 2007 IEEE, DOI 10.1109/ADCOM.2007.123.*
7. *Jose Nunez-Yanez, "Energy Proportional Neural Network Inference with Adaptive Voltage and Frequency Scaling", DOI 10.1109/TC.2018.2879333, IEEE Transactions on Computers.*
8. *Arindam Mallik, Gokhan Memik, Peter Dinda, Robert P. Dick, "User-Driven Frequency Scaling", IEEE Computer Architecture Letters Vol. 5, 2006.*
9. *Ming-Feng Chang, Wen-Yew Liang, "Learning-Directed Dynamic Voltage and Frequency Scaling for Computation Time Prediction", 2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11.*
10. *Bishop Brock and Karthick Rajamani, "Dynamic Power Management for Embedded Systems", Published in the Proceedings of the IEEE International SOC Conference, Ðeptember 17 - 20, 2003.*
11. *Fei Wu, Shiyao Jin, Yuming Wang, "A Simple Model for the Energy-Efficient Optimal Real-Time Multiprocessor Scheduling", 978-1-4673-0089-6/12/ ©2012 IEEE.*
12. *G. Levin, S. Funk, C. Sadowski, I. Pye and S. Brandt. "DP-FAIR A Simple Model for Understanding Optimal Multiprocessor Scheduling", In Proc. of the 22th Euromicro Conference on Real-Time Systems, 2010.*
13. *H. Cho, B. Ravindran, and E.D. Jensen. "An Optimal Real-time Scheduling Algorithm for Multiprocessor". In Proceedings the 27th IEEE Real-Time System Symposium (RTSS), Los Alamitos, CA, pp. 101–110, 2006.*

14. *S. Funk and V. Nadadur "LRE-TL: An Optimal Multiprocessor Algorithm for Sporadic Task Sets", Conference on Real-Time and Network Systems (RTNS), 2009.*
15. *S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. Varvel, "Proportionate Progress: A Notion of Fairness in Resource Allocation", Algorithmica, 15(6):600–625, 1996.*
16. *Alexei Colin · Arvind Kandhalu Ragunathan (Raj) Rajkumar, "Energy-Efficient Allocation of Real-Time Applications onto Single-ISA Heterogeneous Multi-Core Processors", © Springer Science+Business Media New York 2015.*
17. *Georgios L. Stavrinides and Helen D. Karatza, "Energy-Aware Scheduling of Real-Time Workflow Applications in Clouds Utilizing DVFS and Approximate Computations", 2018 IEEE 6th International Conference on Future Internet of Things and Cloud.*
18. *R. N. Calheiros and R. Buyya, "Energy-efficient scheduling of urgent bag-of-tasks applications in clouds through DVFS", Proceedings of the 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom'14), Dec. 2014, pp. 342–349.*
19. *Alexei Colin, Arvind Kandhalu, Ragunathan (Raj) Rajkumar, "Energy-Efficient Allocation of Real-Time Applications onto Heterogeneous Processors", 2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications.*
20. *C. L. Liu and J. W. Layland, "Scheduling algorithms for Multiprogramming in a hard-real-time environment", Journal of the ACM, vol. 20, no. 1, p. 46-6 1, Jan. 1973.*
21. *J. Y. Leung and J. Whitehead, "On the complexity of fixed-priority scheduling of periodic, real-time tasks", Performance Evaluation, vol. 2, no. 4, pp. 237-250, Dec. 1982.*
22. *Marilyn Wolf, "Advanced Configuration and Power Interface", High-Performance Embedded Computing (Second Edition), 2014.*
23. *Gaikwad, Nikita, Yogita Mistry, and Kiran Inamdar. "Design and Implementation of Energy Efficient Environment Monitoring System." International Journal of Electronics, Communication & Instrumentation Engineering Research and Development (IJECIERD) ISSN (P) (2016).*
24. *Patil, Bhagyashree, and Maruti Limkar. "Machine to machine communication based electricity monitoring and billing system." International Journal of Electrical and Electronics Engineering Research (IJEEER) ISSN (P) (2016).*
25. *Ragavendran, U., and M. Ramachandran. "Multistage Current Starved Vco For Switching Applications." International Journal of Mechanical and Production Engineering Research and Development (IJMPERD) Special 2018: 42-50.*
26. *Verma, Pooja, and Rachna Manchanda. "Design Of Radix-4 Booth Multiplier Using Mgdi And Ptl Techniques." Journal of Electronics, Communication & Instrumentation Engineering Research and Development (IJECIERD) (2014): 9-16.*